**KENNESAW STATE UNIVERSITY**

**College of Computer Science and Software Engineering**

**Department of Computer Science**

**CS 4308: Concepts of Programming Languages 01**

**Project Deliverable 1: Scanner**

**Noah Trinite:** [**ntrinite@students.kennesaw.edu**](mailto:ntrinite@students.kennesaw.edu)

**Steven Mayo:** [**smayo13@students.kennesaw.edu**](mailto:smayo13@students.kennesaw.edu)

**George Tatge:** [**gtatge@students.kennesaw.edu**](mailto:gtatge@students.kennesaw.edu)

**October 7, 2020**

The goal of this part of the project is to create a Scanner, may also be called a Lexical Analyzer, to accept and recognize the grammar from a subset of the programming language Ada.

The first major problem we had as a team was having no idea where to even start with this project, but after breaking it down some and looking up more about what a Scanner is we found some great material to help us with the assignment. Rply is a library that is commonly used to make simple compilers/interpreters. With rply we can store our tokens in a LexicalGenerator() object. A string, or strings, can be fed into the LexicalGenerator object, which will check to see if any of the defined tokens appear. Tokens in rply are defined using regular expressions. A problem we ran into when defining our tokens was trying to differentiate between an integer and a float. To overcome this, we had to use particularly complicated regular expressions which gave Noah such a headache he almost gave up and had integers and floats be both defined by “NUMBER.” However, after a little bit more work he figured it out and realized he was overthinking the whole time.

Learning the rply library wasn’t challenging at all; the documentation provided enough examples for us to understand how to implement a scanner pretty easily. After we defined all of our tokens, (which took a while because the more Ada we learned the more we had to add) we then started making an input file to pass into our generator. Learning a new language was rather difficult since Ada has many differences to all the languages we knew beforehand. Steven took on that task and learned enough of this new language, with all of its wacky and different syntax, to supply us with a functioning Ada input file for our scanner in the form of a text file. He made sure it worked by writing it in an online compiler, running it, and then copying it into input.txt. We then pass this text file into a file variable in our python code and iterate through the file variable using our scanner to determine what each token was and what line number the token was on.

There weren’t many limitations for this first deliverable, mainly just us having to decide what order we defined our tokens. If a certain token was defined before another, the scanner might get the intended token confused with another defined token. For example, if we defined the greater than symbol ( > ) before the greater than or equal to symbol ( >= ) then when we pass in the input file and the scanner gets to the >= symbol, it will read the > and = separately. This wasn’t too big of a deal since we got it to work but in a professional environment we would probably spend more time trying to figure out how to get it to differentiate >= and > no matter the placement of the definition within the code.

**Text

Description automatically generatedINPUT FILE:**

**Text

Description automatically generatedDriver Source Code**

**Text

Description automatically generatedScanner Source**

**A screenshot of text

Description automatically generated**

**Text

Description automatically generatedText

Description automatically generatedA close up of text on the side of the water

Description automatically generatedOUTPUT**

**REFERENCES**

RPLY. (n.d.). Retrieved October 07, 2020, from <https://rply.readthedocs.io/en/latest/>

Ada Documentation. (n.d.). Retrieved October 07, 2020, from https://www.adacore.com/documentation

Toarca, S. (n.d.). Python Regular Expression Cheatsheet - Debuggex. Retrieved October 07, 2020, from <https://www.debuggex.com/cheatsheet/regex/python>

Ada Online Compiler. (n.d.). Retrieved October 07, 2020, from <https://rextester.com/l/ada_online_compiler>